🌩️ SPRING CLOUD — COMPLETE INTERVIEW GUIDE (2025 VERSION)

1️⃣ Core Concepts

Q1. What is Spring Cloud and why do we use it?

Spring Cloud is a framework built on Spring Boot that simplifies building **distributed systems and microservices**. It provides:

- Centralized configuration (Spring Cloud Config)
- Service discovery (Eureka)
- Load balancing (Spring Cloud LoadBalancer)
- Circuit breakers (Resilience4j)
- API gateway (Spring Cloud Gateway)
- Distributed tracing (Sleuth + Zipkin)
- Messaging (Spring Cloud Stream with Kafka/RabbitMQ)

🧠 **Example:**

If you have multiple microservices (Order, Payment, Inventory), Spring Cloud manages how they **discover**, **communicate**, and **handle failures**.

2️⃣ Spring Cloud Config

Q2. What is Spring Cloud Config Server?

- Central place to manage **externalized configuration** for all microservices.
- Config files (YAML/properties) are stored in Git or local file system.
- Each service fetches config from it using bootstrap properties.

Example config:

```
spring:
  cloud:
    config:
      uri: http://localhost:8888
      name: inventory-service
```

Q3. How to refresh config without restarting?

- Use @RefreshScope on beans
- Call /actuator/refresh endpoint after config update

Q4. What happens if Config Server is down?

- Services use **last known good configuration** from local cache.
- You can enable **fail-fast** and **retry**.

3️⃣ Service Discovery — Netflix Eureka

Q5. What is Eureka Server?

- A registry where all microservices register themselves.
- Other services use it to find instances by name.

Q6. What are Eureka Client properties?

```
eureka:
  client:
    register-with-eureka: true
```

```
  fetch-registry: true
  service-url:
    defaultZone: http://localhost:8761/eureka/
```

Q7. Difference between @EnableEurekaServer and @EnableEurekaClient?

- @EnableEurekaServer: used in the registry server.
- @EnableEurekaClient: used in each microservice.

Q8. What's the difference between Eureka and Consul/Zookeeper?

- Eureka: Java-based, Netflix OSS, integrated with Spring.
- Consul/Zookeeper: platform-neutral, supports more service health checks.

4️⃣ Load Balancing

Q9. What is Ribbon and why is it deprecated?

- Ribbon was a client-side load balancer by Netflix.
- It's **deprecated** — replaced by **Spring Cloud LoadBalancer**.

Q10. How does Spring Cloud LoadBalancer work?

- Uses service discovery to pick an instance.
- Balances requests between instances.

Example:

```
@LoadBalanced
@Bean
RestTemplate restTemplate() {
  return new RestTemplate();
}
```

5️⃣ API Gateway

Q11. What is Spring Cloud Gateway?

- Gateway for routing requests to downstream services.
- Can perform **routing, filtering, authentication, and rate limiting**.

Q12. Example route configuration:

```
spring:
 cloud:
  gateway:
   routes:
   - id: inventory_route
     uri: lb://inventory-service
     predicates:
       - Path=/inventory/**
     filters:
       - AddRequestHeader=X-Request-Source, Gateway
```

Q13. How does Gateway integrate with Eureka?

- Uses lb://servicename to auto-discover service instances via Eureka.

6️⃣ Circuit Breaker & Resilience

Q14. What is a Circuit Breaker?

- Prevents cascading failures in microservices by “cutting off” a failing service temporarily.

Q15. What libraries are used now?

- Old: Netflix Hystrix (now deprecated)
- New: **Resilience4j**

Q16. Example Resilience4j config:

```
resilience4j.circuitbreaker:
  instances:
    inventoryService:
      slidingWindowSize: 10
      failureRateThreshold: 50
```

Q17. How to use fallback:

```
@CircuitBreaker(name = "inventoryService", fallbackMethod = "fallbackResponse")
public String callInventory() {
   return restTemplate.getForObject("http://inventory-service/status", String.class);
}
```

7️⃣ Distributed Tracing

Q18. What is Spring Cloud Sleuth?

- Adds trace IDs and span IDs to logs for distributed tracing.
- Works with Zipkin or OpenTelemetry.

Q19. Why tracing is important?

- Helps trace requests across multiple services (e.g., “Order → Payment → Inventory”).

8️⃣ Spring Cloud Bus

Q20. What is Spring Cloud Bus used for?

- Connects distributed services using a lightweight message broker (Kafka/RabbitMQ).
- Propagates config changes automatically across all instances.

9️⃣ Spring Cloud Stream

Q21. What is Spring Cloud Stream?

- A framework for building event-driven microservices.
- Abstracts Kafka/RabbitMQ with simple @StreamListener or functional model.

Q22. Example (Kafka binder):

```
@Bean
public Consumer<String> processOrder() {
    return message -> System.out.println("Processing: " + message);
}
```

🔟 Security in Microservices

Q23. How to secure microservices in Spring Cloud?

- Use **OAuth2 / JWT** for authentication.
- Spring Cloud Gateway often acts as a **token validator**.

Q24. What's the role of Keycloak / Auth0 / Okta?

- External Identity Providers (IdPs) for centralized authentication.

1️⃣1️⃣ Version Compatibility & Trends (2025)

Feature	Old	New (2025)
Circuit Breaker	Hystrix	Resilience4j
Load Balancer	Ribbon	Spring Cloud LoadBalancer
Service Registry	Eureka	Eureka / Consul
Gateway	Zuul	Spring Cloud Gateway
Tracing	Sleuth + Zipkin	OpenTelemetry support
Config Reload	Actuator Refresh	Spring Cloud Bus
Message Streams	@StreamListener	Functional Model (Supplier/Consumer)

1️⃣2️⃣ Real-World / Scenario Questions

Q25. How do you handle configuration across environments (dev, test, prod)?

→ Separate YAMLs in Git, fetched by Config Server based on profile.

Q26. How do you make your microservice resilient?

→ Retry, Circuit Breaker, Bulkhead, Timeout, Fallback using Resilience4j.

Q27. How do services communicate securely?

→ HTTPS + OAuth2 tokens between microservices + Gateway validation.

Q28. How do you trace an issue across multiple microservices?

→ Use Sleuth/OpenTelemetry to correlate logs using trace IDs.

Q29. How do you update configs dynamically across all microservices?

→ Change config in Git → Commit → Bus Refresh → All services updated.

1️⃣3️⃣ Important Annotations (Quick Sheet)

Annotation	Purpose
@EnableEurekaServer	Start service registry
@EnableEurekaClient	Register microservice
@EnableConfigServer	Start config server
@EnableCircuitBreaker	(Old, Hystrix)
@RefreshScope	Refresh config dynamically
@LoadBalanced	Enable client-side load balancing
@FeignClient	Declarative REST client
@EnableFeignClients	Enable Feign scanning

---

1 ☸️ Spring Cloud with Docker & Kubernetes

Q30. How does Spring Cloud work with Kubernetes?

- You can replace Eureka with **Kubernetes service discovery**.
- Config handled via **ConfigMap** or **Spring Cloud Kubernetes Config**.

Q31. Is Spring Cloud still needed on K8s?

- Partially — K8s already offers discovery, config, and scaling.
- Spring Cloud adds **resilience, circuit breaker, and distributed tracing**.
